MATLAB PROJECT 3

Please include this page in your Group file, as a front page. Type in the group number and the names of all members WHO PARTICIPATED in this project.

GROUP # \_\_\_\_13\_\_\_\_\_\_\_

FIRST & LAST NAMES (UFID numbers are NOT required):

\_1.\_\_\_\_\_\_\_\_\_\_\_\_\_Sean O’Reilly\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_Keval Patel\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ­­­

\_3.\_\_\_\_\_\_\_\_\_\_\_\_\_Nicholas Miller\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_4.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Carl Simpron\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_5.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Daniel Silva\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_6.\_\_\_\_\_\_\_\_\_\_\_\_\_Koby Miller\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**By signing your names above, each of you had confirmed that you did the work and agree with the work submitted**.

format compact

diary on

%Exercise #1

type subspace

function[] = subspace(A,B)

m=size(A,1);

n=size(B,1);

if m~=n

sprintf('Col A and Col B are subspaces of different spaces')

return

else

sprintf('Col A and Col B are subspaces of R^%i\n',m)

end

k = rank(A);

p = rank(B);

fprintf('dim of Col A is k = %i\n',k)

fprintf('dim of Col B is p = %i\n',p)

if(k ~= p)

disp('k ~= p, the dimensions of Col A and Col B are different')

else

C = rref(A.');

D = rref(B.');

if(C == D)

disp('Col A = Col B')

else

disp('k = p, the dimensions of Col A and Col B are the same, but Col A ~ = Col B')

end

end

if(k == m)

fprintf('k = m (%i=%i) Col A is all R^%i\n',k,m,m)

else

fprintf('k ~= m (%i~=%i) Col A is not all R^%i\n',k,m,m)

end

if(p == m)

fprintf('p = m (%i=%i) Col B is all R^%i\n',p,m,m)

else

fprintf('k ~= m (%i~=%i) Col B is not all R^%i\n',p,m,m)

end

end

%(a)

A = [2 -4 -2 3; 6 -9 -5 8; 2 -7 -3 9; 4 -2 -2 1; -6 3 3 4]

A =

2 -4 -2 3

6 -9 -5 8

2 -7 -3 9

4 -2 -2 1

-6 3 3 4

B = rref(A)

B =

1.0000 0 -0.3333 0

0 1.0000 0.3333 0

0 0 0 1.0000

0 0 0 0

0 0 0 0

subspace(A,B)

ans =

'Col A and Col B are subspaces of R^5

'

dim of Col A is k = 3

dim of Col B is p = 3

k = p, the dimensions of Col A and Col B are the same, but Col A ~ = Col B

k ~= m (3~=5) Col A is not all R^5

k ~= m (3~=5) Col B is not all R^5

%(b)

A = magic(4)

A =

16 2 3 13

5 11 10 8

9 7 6 12

4 14 15 1

B = eye(4)

B =

1 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1

subspace(A,B)

ans =

'Col A and Col B are subspaces of R^4

'

dim of Col A is k = 3

dim of Col B is p = 4

k ~= p, the dimensions of Col A and Col B are different

k ~= m (3~=4) Col A is not all R^4

p = m (4=4) Col B is all R^4

%(c)

A = magic(4)

A =

16 2 3 13

5 11 10 8

9 7 6 12

4 14 15 1

B = eye(3)

B =

1 0 0

0 1 0

0 0 1

subspace(A,B)

ans =

'Col A and Col B are subspaces of different spaces'

%(d)

A=magic(5)

A =

17 24 1 8 15

23 5 7 14 16

4 6 13 20 22

10 12 19 21 3

11 18 25 2 9

B = eye(5)

B =

1 0 0 0 0

0 1 0 0 0

0 0 1 0 0

0 0 0 1 0

0 0 0 0 1

subspace(A,B)

ans =

'Col A and Col B are subspaces of R^5

'

dim of Col A is k = 5

dim of Col B is p = 5

Col A = Col B

k = m (5=5) Col A is all R^5

p = m (5=5) Col B is all R^5

%A side effect of row reduction is the rounding of decimals like what happened to what

%should have been 1/3. This is why the function did not work on A

%Exercise #2

type shrink

function B = shrink(A)

[~,pivot] = rref(A);

B = A(:,pivot);

end

A = magic(4)

A =

16 2 3 13

5 11 10 8

9 7 6 12

4 14 15 1

%Creates a 4 by 4 matrix whose diagnols, columns and row add up to 16

[~,pivot]=rref(A)

pivot =

1 2 3

%This gives the pivot columns of the row reduced matrix A

B = A(:,pivot)

B =

16 2 3

5 11 10

9 7 6

4 14 15

%This makes a matrix of the pivot columns of A.

type basis

function B = basis(A)

%I took out the \n because it makes the output look better in my opinion

m = size(A,1);

A = shrink(A);

sprintf('a basis for ColA is')

B = A

if rank(B) == size(A,1)

sprintf('a basis for R^%i is',m)

else

D = [B eye(m)];

D = shrink(D);

if rank(D) == size(A,1)

sprintf('a basis for R^%i is',m)

B = D;

else

disp('What? It is not a basis?!')

end

end

end

%(a)

A = [1 0;0 0;0 0;0 1]

A =

1 0

0 0

0 0

0 1

B = basis(A)

ans =

'a basis for ColA is'

B =

1 0

0 0

0 0

0 1

ans =

'a basis for R^4 is'

B =

1 0 0 0

0 0 1 0

0 0 0 1

0 1 0 0

%(b)

A = [2 0;4 0;1 0;0 0]

A =

2 0

4 0

1 0

0 0

B = basis(A)

ans =

'a basis for ColA is'

B =

2

4

1

0

ans =

'a basis for R^4 is'

B =

2 1 0 0

4 0 1 0

1 0 0 0

0 0 0 1

%(c)

A = magic(3)

A =

8 1 6

3 5 7

4 9 2

B = basis(A)

ans =

'a basis for ColA is'

B =

8 1 6

3 5 7

4 9 2

ans =

'a basis for R^3 is'

B =

8 1 6

3 5 7

4 9 2

%(d)

A=magic(6)

A =

35 1 6 26 19 24

3 32 7 21 23 25

31 9 2 22 27 20

8 28 33 17 10 15

30 5 34 12 14 16

4 36 29 13 18 11

B = basis(A)

ans =

'a basis for ColA is'

B =

35 1 6 26 19

3 32 7 21 23

31 9 2 22 27

8 28 33 17 10

30 5 34 12 14

4 36 29 13 18

ans =

'a basis for R^6 is'

B =

35 1 6 26 19 1

3 32 7 21 23 0

31 9 2 22 27 0

8 28 33 17 10 0

30 5 34 12 14 0

4 36 29 13 18 0

%exercise 3

format compact

syms x

%(a)

B=[x^3+3\*x^2,10^(-8)\*x^3+x,10^(-8)\*x^3+4\*x^2+x,x^3+x]

B =

[ x^3 + 3\*x^2, x^3/100000000 + x, x^3/100000000 + 4\*x^2 + x, x^3 + x]

Q=10^(-8)\*x^3+x^2+6\*x

Q =

x^3/100000000 + x^2 + 6\*x

r=[2;-3;1;0]

r =

2

-3

1

0

P = polyspace(B,Q,r)

ans =

'the polynomials in B do not form a basis for P3'

the reduced echelon form of P is P =

1 0 0 1

3 0 4 0

0 1 1 1

0 0 0 0

%this is not a basis for P3 due to a lack of pivot positions

%(b)

B=[x^3-1,10^(-8)\*x^3+2\*x^2,10^(-8)\*x^3+x,x^3+x]

B =

[ x^3 - 1, x^3/100000000 + 2\*x^2, x^3/100000000 + x, x^3 + x]

P = polyspace(B,Q,r)

ans =

'the polynomials in B form a basis for P3'

the coordinates of the polynomial Q with respect to the basis P are y =

0 1/2 6 0

the coordinate vector q of the polynomial R with respect to the standard

basis is q =

2 -6 1 -2

the polynomial R is R =

2\*x^3 - 6\*x^2 + x - 2

P =

1 0 0 1

0 2 0 0

0 0 1 1

-1 0 0 0

%C

B=[x^4+x^3+x^2+1,10^(-8)\*x^4+x^3+x^2+x+1,10^(-8)\*x^4+x^2+x+1, 10^(-

8)\*x^4+x+1,10^(-8)\*x^4+1]

B =

[ x^4 + x^3 + x^2 + 1, x^4/100000000 + x^3 + x^2 + x + 1, x^4/100000000 + x^2

+ x + 1, x^4/100000000 + x + 1, x^4/100000000 + 1]

Q=x^4-1

Q =

x^4 - 1

r=diag(magic(5))

r =

17

5

13

21

9

P = polyspace(B,Q,r)

ans =

'the polynomials in B form a basis for P4'

the coordinates of the polynomial Q with respect to the basis P are y =

1 -1 0 1 -2

the coordinate vector q of the polynomial R with respect to the standard

basis is q =

17 22 35 39 65

the polynomial R is R =

17\*x^4 + 22\*x^3 + 35\*x^2 + 39\*x + 65

P =

1 0 0 0 0

1 1 0 0 0

1 1 1 0 0

0 1 1 1 0

1 1 1 1 1

% Exercise 4

type reimsum

function [T,I] = reimsum(P,a,b,n)

format long;

pcv = sym2poly(P); %Polynomial Coefficient Vector

[p,q] = size(n);

sum = 0;

for j=1:q

length = (b - a) / n(1,j);

leftSum = a;

rightSum = a + length;

middleSum = a + (length / 2);

sum = 0;

for i=1:n(1,j) %Left Riemann Sum

sum = sum + (length \* (polyval(pcv, leftSum)));

leftSum = leftSum + length;

end

c(1,j) = sum;

sum = 0;

for i=1:n(1,j) %Middle Riemann Sum

sum = sum + (length \* (polyval(pcv,middleSum)));

middleSum = middleSum + length;

end

d(1,j) = sum;

sum = 0;

for i=1:n(1,j) %Right Riemann Sum

sum = sum + (length \* (polyval(pcv, rightSum)));

rightSum = rightSum + length;

end

f(1,j) = sum;

end

d = closetozeroroundoff(d);

A = [transpose(n) transpose(c) transpose(d) transpose(f)];

T = array2table(A,...

'VariableNames',{'n','left','Middle','Right'});

T = evalc('disp(T,false)'); %Removes html tags present in the table

I = double(int(P,a,b));

syms x

% (a)

P = (2\*x^4) + (4\*x^2) - 1;

a=-1; b=1;

n = [1:10];

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 10 -2 10

2 4 0.25 4

3 2.62551440329218 0.897119341563786 2.62551440329218

4 2.125 1.140625 2.125

5 1.88992 1.25632 1.88992

6 1.76131687242798 1.31995884773663 1.76131687242798

7 1.68346522282382 1.35860058309038 1.68346522282382

8 1.6328125 1.3837890625 1.6328125

9 1.5980287557791 1.40110755474267 1.5980287557791

10 1.57312 1.41352 1.57312

'

I =

1.466666666666667

n = [1,5,10,100,1000,10000]

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 10 -2 10

5 1.88992 1.25632 1.88992

10 1.57312 1.41352 1.57312

100 1.467733312 1.466133352 1.467733312

1000 1.46667733333121 1.46666133333521 1.46667733333121

10000 1.46666677333278 1.46666661333278 1.46666677333278

'

I =

1.466666666666667

a=-10; b=10;

n = [1:10];

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 407980 -20 407980

2 203980 26980 203980

3 139864.773662551 55025.2674897119 139864.773662551

4 115480 66542.5 115480

5 103852 72172 103852

6 97445.0205761317 75309.2181069959 97445.0205761317

7 93551.0120783007 77227.8134110787 93551.0120783008

8 91011.25 78483.90625 91011.25

9 89264.3570593914 79350.0147335264 89264.3570593914

10 88012 79972 88012

'

I =

8.264666666666667e+04

n = [1,5,10,100,1000,10000];

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 407980 -20 407980

5 103852 72172 103852

10 88012 79972 88012

100 82700.5312000001 82619.7352000001 82700.5312000001

1000 82647.2053331168 82646.3973335166 82647.2053331167

10000 82646.6720533182 82646.6639733178 82646.6720533182

'

I =

8.264666666666667e+04

% (b)

P = x^3 - (2\*x);

a=-1; b=1;

n = [1:10];

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 2 0 -2

2 1 0 -1

3 0.666666666666667 0 -0.666666666666667

4 0.5 0 -0.5

5 0.4 0 -0.4

6 0.333333333333333 0 -0.333333333333333

7 0.285714285714286 0 -0.285714285714286

8 0.25 0 -0.25

9 0.222222222222222 0 -0.222222222222222

10 0.2 0 -0.2

'

I =

0

n = [1,5,10,100,1000,10000];

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 2 0 -2

5 0.4 0 -0.4

10 0.2 0 -0.2

100 0.019999999999999 0 -0.0200000000000009

1000 0.00199999999999853 0 -0.00200000000000131

10000 0.000200000000151129 0 -0.000199999999848991

'

I =

0

a=-10; b=10;

n = [1:10];

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 -19600 0 19600

2 -9800 0 9800

3 -6533.33333333333 0 6533.33333333333

4 -4900 0 4900

5 -3920 0 3920

6 -3266.66666666667 0 3266.66666666667

7 -2800 0 2800

8 -2450 0 2450

9 -2177.77777777778 0 2177.77777777778

10 -1960 0 1960

'

I =

0

n = [1,5,10,100,1000,10000];

[T,I] = reimsum(P,a,b,n)

T =

' n left Middle Right

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 -19600 0 19600

5 -3920 0 3920

10 -1960 0 1960

100 -196.000000000004 0 195.999999999996

1000 -19.6000000002456 0 19.5999999997534

10000 -1.95999999998541 0 1.96000000001435

'

I =

0

% Left and Right Riemann Sums provided very similar values for any given

value n(1,j).

% The only difference between the two is that for part (b) they both had

opposite signs.

% Left and Right for n[1:10] also provided inaccurate results for the

integral.

% However for n[1,5,10,100,1000,10000], Left and Right provided decent

approximations for the integral at about when n > 1000.

% Middle Riemann Sums, though, provided very accurate results for the

integral.

% Even when n = [1:10], Middle gets closer to the actual integral of the

equation than Left and Right.

% Therefore, Middle Riemann Sum provides a better approximation for the

integral than both Left and Right Riemann Sums.

%Part 5

type polint

function B = polint(P)

syms x;

u = sym2poly(P);

len = length(u);

answer = [u,0]

for i = 1:len

answer(i) = answer(i)/(len+1-i);

end

poly2sym(answer)

end

polint(6\*x^5+5\*x^4+4\*x^3+3\*x^2+2\*x+6)

answer =

6 5 4 3 2 6 0

ans =

x^6 + x^5 + x^4 + x^3 + x^2 + 6\*x

int(6\*x^5+5\*x^4+4\*x^3+3\*x^2+2\*x+6)

ans =

x^6 + x^5 + x^4 + x^3 + x^2 + 6\*x

polint(x^6-x^4+3\*x^2+1)

answer =

1 0 -1 0 3 0 1 0

ans =

x^7/7 - x^5/5 + x^3 + x

int(x^6-x^4+3\*x^2+1)

ans =

x^7/7 - x^5/5 + x^3 + x

%Exercise 6

type markov

function q = markov(P, x0)

S1 = sum(P,1); %sum of columns

S2 = (sum(P,2))'; %sum of rows

[r,c] = size(P); %Used to see if the matrix is a square, also used to

%traverse the matrix when needed

if any(S1==0)==1 && any(S2==0)==1 %This checks if there is a row AND

%and column that are all 0's

disp('P is not a stochastic matrix')

q = [];

return

end

if any(P(:)<0) == 1 %checks for negative values

disp('P is not a stochastic matrix')

q = [];

return

end

if r ~= c %checks if the matrix is square

disp('P is not a stochastic matrix')

q = [];

return

end

if any(S1(:)~= 1) %checks if the values in each col

%adds up to 1

disp('P is not a stochastic matrix')

q = [];

return

end

S3 = sum(x0,1); %sum of col in x0 vector

if(S3~=1)

disp('x0 is not a probability vector')

q = [];

return

end

if any(x0(:)<0) == 1 %checks for negative values

disp('P is not a stochastic matrix')

q = [];

return

end

Q = null(P-eye(r), 'r');

C = sum(Q);

q = Q/C;

k=0; %This will be the counter for vector x

x=x0;

i=0; %this will be used to control the while loop

if(q == x0) %if they are already equal, don't go through the loop

i = 1;

end

while(i == 0)

x = P\*x; %this is the x\_n = Px\_n+1

if(norm(x - q)< 10^(-7))

i = 1;

end

k=k+1; %keep track of iteration

end

fprintf('\nThe number of iterations k was: %g',k)

fprintf('\nx%g',k)

fprintf(' = \n\n')

disp(x)

end

%(a)

P = [.6 .3; .5 .7]

P =

0.6000 0.3000

0.5000 0.7000

x0 = [.4;.6]

x0 =

0.4000

0.6000

q = markov(P,x0)

P is not a stochastic matrix

q =

[]

%(b)

P = [.5 .3; .5 .7]

P =

0.5000 0.3000

0.5000 0.7000

x0 = [.4;.6]

x0 =

0.4000

0.6000

q = markov(P,x0)

The number of iterations k was: 8

x8 =

0.3750

0.6250

q =

0.3750

0.6250

%x8 and q are the same

%(c)

P = [.9 .2; .1 .8]

P =

0.9000 0.2000

0.1000 0.8000

x0 = [.12;.88]

x0 =

0.1200

0.8800

q = markov(P,x0)

The number of iterations k was: 45

x45 =

0.6667

0.3333

q =

0.6667

0.3333

%(d)

%part1

P = [.9 .2; .1 .8]

P =

0.9000 0.2000

0.1000 0.8000

x0 = [.14;.86]

x0 =

0.1400

0.8600

q = markov(P,x0)

The number of iterations k was: 45

x45 =

0.6667

0.3333

q =

0.6667

0.3333

%part2

P = [.9 .2; .1 .8]

P =

0.9000 0.2000

0.1000 0.8000

x0 = [.86;.14]

x0 =

0.8600

0.1400

q = markov(P,x0)

The number of iterations k was: 42

x42 =

0.6667

0.3333

q =

0.6667

0.3333

%The q for part (c) and both parts of (d) were the same. The initial vector

%x0 only effected the amount of iterations it took to reach the

%steady-state vector. In part (c) it took 45 but int the second part of (d)

%it only took 42.

%(e)

P = [.90 .01 .09; .01 .90 .01; .09 .09 .90]

P =

0.9000 0.0100 0.0900

0.0100 0.9000 0.0100

0.0900 0.0900 0.9000

x0 = [.5;.3;.2]

x0 =

0.5000

0.3000

0.2000

q = markov(P,x0)

The number of iterations k was: 128

x128 =

0.4354

0.0909

0.4737

q =

0.4354

0.0909

0.4737

diary off